Taxonomy of Low-level Hybridization (LLH) for PSO-GA

S.Masrom, Siti Z.Z. Abidin, N.Omar, and K.Nasir

***Abstract*— Particle Swarm Optimization (PSO) is a popular algorithm used extensively in continuous optimization. One of its well-known drawbacks is its propensity for premature convergence. Many techniques have been proposed for alleviating this problem. One of the popular and promising approaches is low-level hybridization (LLH) of PSO with Genetic Algorithm (GA). Nevertheless, the LLH implementation is considerably difficult due to internal structure modifications of the original hybrid algorithms. Many success works have been reported on LLH for PSO-GA but a wide range of presumption terms and terminology are used. This paper describes the numerous techniques of LLH for PSO-GA in a form of simple taxonomy. Then, examples of several implementation models based on the taxonomy are given. Recent trends are also briefly discussed from an implementations review.**

***Index Terms*—Meta-heuristics, Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Low-level Hybridization (LLH), Taxonomy**

I. Introduction

From the family of meta-heuristics algorithms, Particle Swarm Optimization (PSO) [1] and Genetic Algorithm (GA) [2] are the two well-known and popular search strategies that have gained widespread appeal amongst researchers to solve optimization problems in a variety of application domains. These algorithms were developed based on nature analogy but have different in several principles. The searching idea of PSO is to mimic social activities of animals such as birds flocking and fish schooling. GA in other ways is simulating natural evolution of creatures such as genetic reproduction and mutation.

Due to the different searching paradigm, each PSO and GA has its own strengths and weaknesses when generating optimal solutions for optimization problems. PSO is known to be very effective in producing fast results but tends to converge to a local optimum [3]. It often has problem with less diversity to explore a wide range of potential solutions in the search space. Therefore, in most cases especially to real life optimization problem, the optimal results produced by PSO are still insufficient.

Manuscript received Jan 8, 2014; revised Jan 30, 2014. This work was supported by the Kementerian Pengajian Tinggi Malaysia and Universiti Teknologi MARA under Grant 600-RMI/FRGS 5/3 (10/2012).

S.Masrom is with the Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Perak, Malaysia (e-mail: suray078@perak.uitm.edu.my).

Siti Z.Z. Abidin is with the Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Shah Alam, Malaysia (e-mail: sitizaleha533@tmsk.uitm.edu.my).

N. Omar is with the Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Shah Alam, Malaysia (e-mail: nasiroh@tmsk.uitm.edu.my).

K. Nasir is with the Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Shah Alam, Malaysia (e-mail: kalsom@tmsk.uitm.edu.my).

On the other hand, GA was generally found to have better search diversity than PSO [4]. Although it is still susceptible to premature convergence like PSO, the search diversity in GA can be controlled with several operators such as mutation and crossover. As a result, GA is very effective in producing accurate results. Nevertheless, GA faces with long processing time from the excessive computational burden of the control operators [5].

An integration of strengths from PSO and GA can yield a new meta-heuristic with better efficiency than the single algorithm. Generally known as *meta-heuristics hybridization,* the combination techniques have been proven to be very effective in solving many kinds of optimization problems [6][7]. Nevertheless, to implement meta-heuristics hybridization is considerable difficult than the single version. While almost every report on meta-heuristics hybridization presents such a success story, attempting to understand the algorithm designs and replicating the experiments appears to be so heavily. Besides, most works provide very brief reports on the hybridization techniques and use a variety of presumption terms and terminology [8].

As to reduce the difficulties, many researchers attempt to provide general and simplified descriptions for different implementations of metaheuristics hybridization by proposing classification or taxonomy. Based on the different taxonomies, researchers have a common view that metaheuristics hybridizations can be generally classified as high-level hybridization (HLH) and low-level hybridization (LLH) [9][6][7].

In HLH, both algorithms interact each other through a well defined interface or protocol and the components from different algorithms are not strongly dependent [10]. Therefore, the algorithms in HLH can be retained with their original identity or algorithm structured.

Different with LLH, the techniques involve internal structure modifications of the hybrid algorithms. In other words, LLH creates new algorithm that combine components from different hybrid algorithms [11]. The components are strongly inter-dependent and must be fit well together. Therefore, an appropriate design and technique for LLH implementation is essential which needs programmer to understand well the structure and working paradigm of the different algorithms. Although several taxonomies are reported to increase users understanding on metaheuristics hybridizations, there are still limited works provided by LLH [12].

II. Related Works

The main idea that classified meta-heuristics hybridization into its hybrid level was originally proposed by Talbi in [9]. He has introduced taxonomy for metaheuristics hybridizations with regards to *high-level*

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

teamwork, high-level relay, low-level teamwork and *low-level relay*. Furthermore, in a more general view, researchers in[6], has divided the hybrid metaheuristics scheme into three general forms which are *component exchange among meta-heuristics*, *cooperative search from different meta-heuristics* and *integration with others methods.* The researchers agree that component exchange among meta-heuristics can be implemented with *high-level* or *low-level* methods.

Parallel metaheuristics are also considered as metaheuristics hybridization due to the occurrence of cooperation among different metaheuristics [6]. There are several taxonomies for parallel metaheuristics have been proposed but it is more significant to HLH. For examples, a taxonomy that categorized the implementation as *operation parallelization, search space decomposition* and *multi-search threads* [13]. In [14], the researchers have classified parallel metaheuristics according to *algorithm types* and *space decomposition*.

There are also taxonomies proposed for a particular metaheuristics type. Some of them are Tabu search [15], Particle Swarm Optimization [16], Ant Colony Optimization [17] and Evolutionary Algorithms (EAs) [8] [18][19][20]. Although these taxonomies are relevant to hybridization, none is found to be specifically proposed for LLH.

Given previously limited works for LLH and with the advantages of PSO-GA hybridization, this paper proposes taxonomy of LLH for PSO-GA. Before the proposed taxonomy is presented, the following part describes the fundamental elements and processes of LLH.

III. General definitions

The LLH of PSO-GA can be formally defined as a composition of *(m,s)* where $m, s \in M$ are the two different algorithms of PSO and GA from a family of meta-heuristics algorithms, $M = \{a_1, a_2..a_n\}$. The parameter *m* and *s* are devoted to *master-metaheristic* and *sub-metaheuristic* respectively. In this study, the focus is given to one-way LLH where PSO is always the *m* and GA as the *s.*

As a meta-heuristic *M*, each PSO and GA consists of general components $G = \{S, f, \Omega\}$ and proprietary components $P = \{x_1, x_2..x_n\}$. The general components are common to all *M* algorithms but they are distinct with proprietary components.

The general component *S* consists of solutions in search space that has a finite set of decision variables V_i where $i = \{1..n\}$. The type of variables can be in discrete, continuous or mixed format, represented with a particular encoding of algorithm *M* [6]. Each PSO and GA can work on identical representations for all solutions in the search space or employs different representations respectively [14]. Therefore, the LLH might be able to operate not just on one search space but also on different search space compositions [12].

The set of constraints among the variables is defined in a set of penalty functions $\Omega = \{f_1, f_2.., f_3\} | f_x : V_i \rightarrow C$ where $i = \{1..n\}$. The objective function $f = S \rightarrow R$ assigns a cost value *MIN* and *MAX* to each solution of *S.*

Proprietary components *P* are exclusive to their respective meta-heuristic *M* which can be crossover into the routine of another metaheuristics [21]. In order to maintain the main paradigm of PSO as a *master-metaheuristic*, the LLH must implement PSO's proprietary components but is option to include one or more GA's proprietary components.

The general and proprietary components have several parameters that can be associated with *dynamic* or *static* value[22][12]. *Static* parameter value is constant along all search iterations while *dynamic* value is changeable according to *self-adaptive* $A = \{a_1, a_2..a_n\}$ or *time-varying* $V = \{v_1, v_2..v_n\}$ behaviour. The sets of *A* and *V* consist of different functions with different types *T* that formulating the *dynamic* behaviour.

IV. The taxonomy

Based on the LLH definition, the taxonomy for LLH of PSO-GA is generally divided into *component* and *implementation* as shown in Fiq. 1.

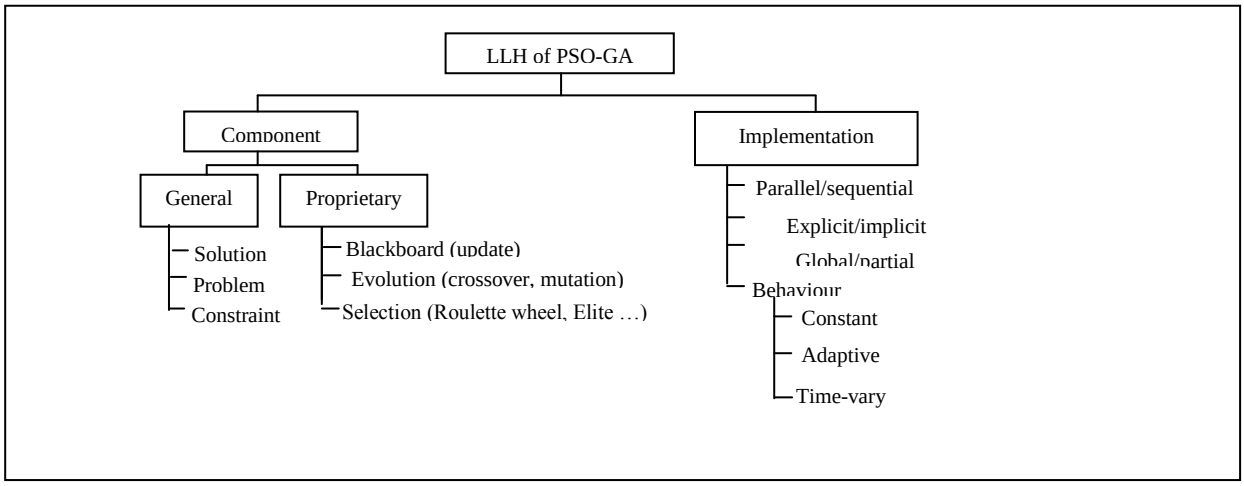

Fig.1. Taxonomy for LLH of PSO-GA

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

A. Components

As meta-heuristics, PSO and GA have general and proprietary components. The general components include *problem* to be solved, a group of *solutions* for the problem and the solutions *constraints*.

The solutions for the problems in the search space are represented according to the particular algorithm. PSO uses *particle* for representing solutions while in GA in the form of *chromosome*. The problem is defined through one or more objective functions while solutions constraints can be derived with constraint functions.

The proprietary consists of specific components for PSO and GA. While PSO proprietary components based on *blackboard* type, GA components comprised of *evolution* and *selection* categories. The *evolution* approach uses some operators (e.g., *mutation* and *crossover*) to reproduce new population of solutions while *blackboard* type is utilizing shared memory concept when generating new population by updating some information of solutions. PSO uses its shared memory in the form of personal and global best.

Selection technique is common to GA algorithm. There are varieties of selection techniques have been introduced into the algorithm including *roulette wheel*, *tournament* and *rank-based*. Another popular method that can also be associated with selection is *elitism* that create new group of best solutions from the current solutions [11].

B. Implementations

Implementation refers to execution method for the LLH components. For example, the solutions component in search space can be composed into several sub-search spaces which can be explored in *parallel* or *sequential.* If encoding method for the solutions representation is identical for each sub-search spaces, it is categorized as *explicit.* Otherwise it is classified as *implicit* decomposition. Further, each algorithm might solve on *global* or *partial* problem. The problem is *global* if both PSO and GA solve the same target optimization problem while *partial* problem occurs if the problem is different for each algorithm.

The *behavior* element refers to parameters value of each parameter of components which can be *constant* or *dynamic.* Formulation of *dynamic* behavior derives either from *random, time-varying* or *adaptive.* The *time-varying* depends majorly on search iteration number while *adaptive* behavior reflects on current performance of algorithm search such as the local or global fitness. Some of available formulations for *time-varying* are *linear increasing (LI), non-linear increasing (NLI)* and *non-linear decreasing*[23].

C. Implementation models

There are ten models of implementation can be applied for the LLH in relation to the *component* and *implementation* classification as shown in Fig. 2. Each method is categorized relatively to *search space exploration (parallel* or *sequential), solution decomposition (explicit* or *implicit)* and *problem (global* or *partial).*

In more details, the following part gives flow chart of some implementation models. Then, in order to illustrates connection between the taxonomy elements (*components* and implementation), the configurations for each model is given in a form of simple statements.

i. Parallel explicit global

As illustrated in Fig. 3, this method divides search spaces into two sub-search spaces. Each PSO and GA is exploring their respective search space in parallel. Since both search spaces are represented with PSO *particle*, the solutions decomposition is defined as *explicit*. Besides, both PSO and GA work on solving the same *global* problem.

Furthermore, Fig. 4 shows extra descriptions for the method that includes behavior characteristics.

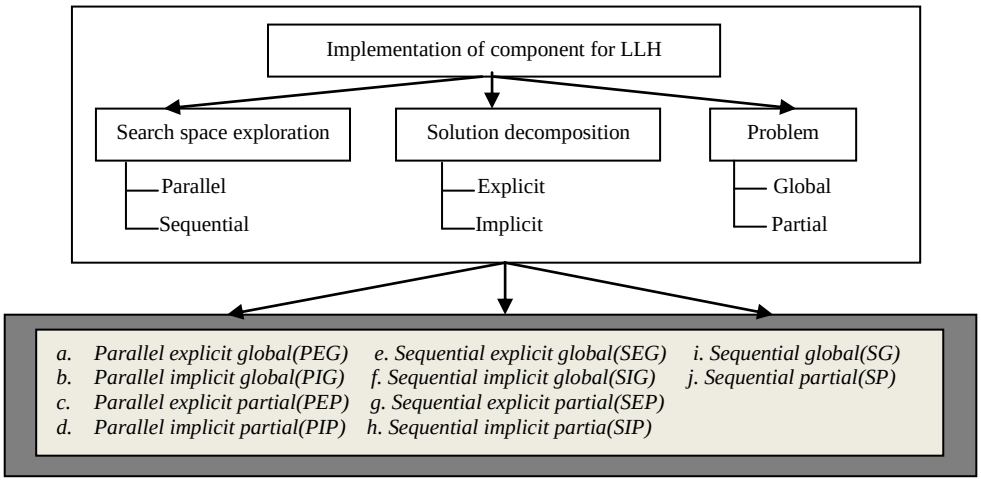

Fig.2. Implementation models

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

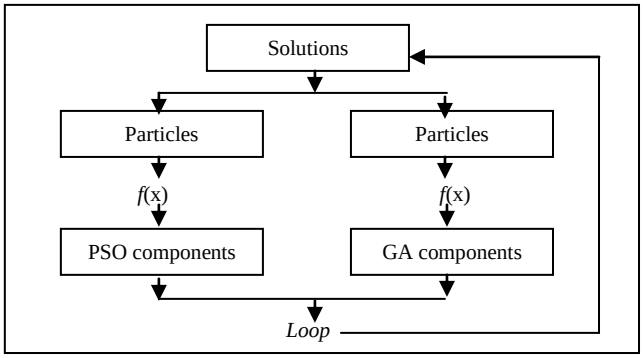

Fig.3. Flow chart for *parallel explicit global*

- Search space (parallel [2, 0.5])
- Solutions (explicit [particle])
-Problem (Global [f(x)])
- Update (inertia [constant]), personal learning [constant], social learning [constant])
- Selection (Elite [constant])
- Crossover (rate [constant], operation [one point, adaptive [bestfitness]])
- Mutation (rate [constant], operation [Gaussian, adaptive [personalfitness]])

Fig.4. Example of configurations for *parallel explicit global*

In the configurations, search space is divided into two sub-search spaces with equal percentage defines with Search space (parallel [2, 0.5]). The behaviour characteristics are configured through the parameters. For example the *adaptive* behavior for *crossover* operation is derived from *bestfitness* formulation while *mutation* operation is dynamically determined with *personalfitness adaptive* factor.

ii. Parallel explicit partial

Similar with *parallel explicit global*, this method divides search spaces into two sub-search spaces where all solutions represented with PSO *particle.* However, the method solves *partial* kind of problem where main problem solved by PSO as the *master-metaheuristic* while sub-problem solved by GA. Fig. 5 shows the flow chart while Fig. 6 gives examples of the configurations.

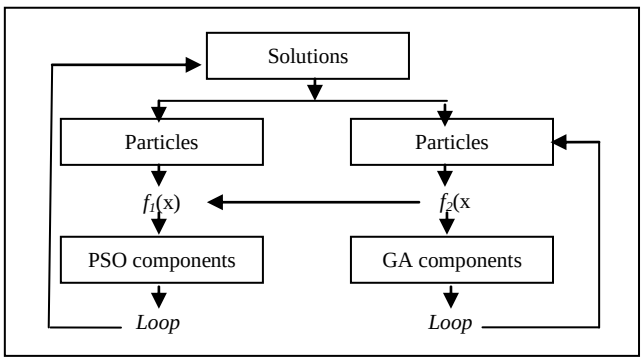

Fig.5. Flow chart for *parallel explicit partial*

- Search space (parallel [2, 0.5])
- Solutions (explicit [particle])
- Problem (Partial [f1(x], [f2(x])
- Update (inertia [time-vary[LD]]), personal learning [constant], social learning [constant])
- Mutation (rate [constant], operation [Gaussian, adaptive [personalfitness]])

Fig. 6.Example of configurations for *parallel explicit partial*

In the configurations, the *adaptive* behaviour for *inertia* parameter is *time-vary* with *linear decreasing (LD)* formulation. Besides, only *mutation* operation from GA components is included to the LLH.

iii. Sequential implicit partial

Different with parallel methods, the sequential method performs search exploration for different search space in a serial manner. As shown

in Fig. 7, solutions in each search space are represented in *particle* and *chromosome* respectively. Thus, it is categorized as *implicit* decomposition.

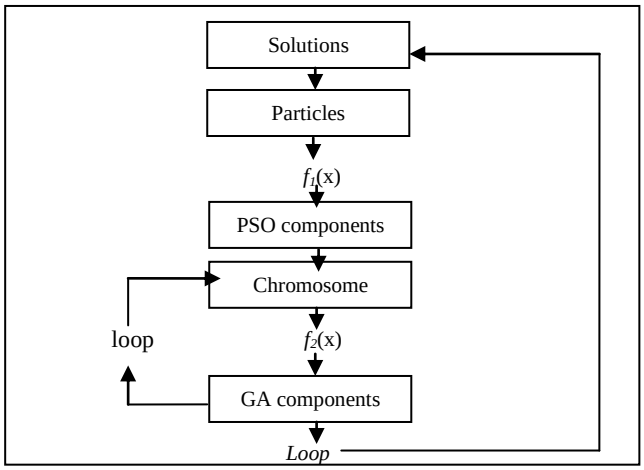

Fig.7. Flow chart for *sequential implicit partial*

The method can be configured as in the Fig. 8 where *mutation rate* is determined according to *time-vary* behavior with *Non-linear decreasing (NLD)* formulation.

- Search space (sequential [2, 0.5])
- Solutions (implicit [particle, chromosome])
- Problem (Partial [f1(x], [f2(x])
- Update (inertia [constant]), personal learning [constant]], social learning [constant])
- Selection (Elite [constant])
- Crossover (rate [constant], operation [one point, adaptive [global fitness]])
- Mutation (rate [time-vary [NLD]], operation [Gaussian, adaptive [best fitness]])

Fig.8 Example of configurations for *sequential implicit partial*

iv. Sequential global

This is an example of sequential implementation without solutions decomposition and thus not applicable for *explicit/implicit* element. Fig. 9 and Fig. 10 show the flow chat and configurations respectively .

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

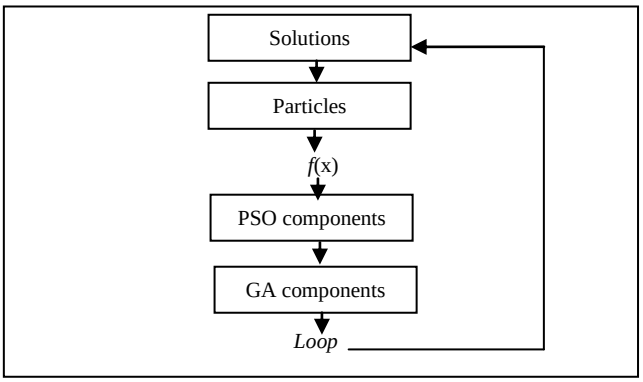

Fig.9. Flow chart for *sequential global*

- Search space (sequential)
- Solutions (explicit [particle])
- Problem (global [f(x])
- Mutation (rate [constant], operation [Gaussian, adaptive [best fitness]])
- Update (inertia [constant]), personal learning [time-vary [LD]]], social learning [time-vary [LD]])

Fig. 10 Example of configurations for *sequential global*

The *selection* and *crossover* components from GA are not included. The *mutation* operation is implemented prior to *update* operation from PSO. This configuration is to show that the implementation of proprietary components and its arrangement is flexible for sequential implementation.

v. Sequential partial

This is another model of sequential implementation with *partial* kind of problem. As seen in Fig. 11, the PSO components are firstly implemented than GA components. This is because PSO is the *master-metaheuristic* and works for main (*global*) problem while GA solves *partial* problem as it acted as *sub-metaheuristic.* Example of configurations for this method can be viewed in Fig. 12.

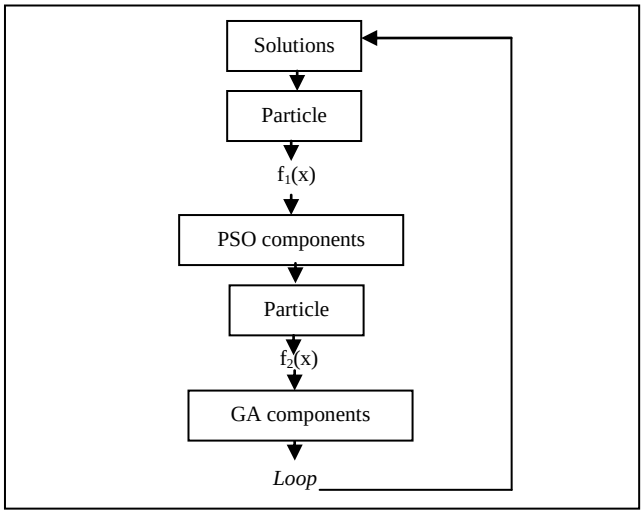

Fig. 11 Flow chart for *sequential partial*

- Search space (sequential)
- Solutions (implicit [particle, chromosome])
- Problem (Partial [f1(x], [f2(x])
- Update (inertia [constant]), personal learning [constant [time-vary [LD]]], social learning [time-vary [LD]])
- Crossover (rate [constant], operation [one point, adaptive [global fitness]])
- Mutation (rate [constant], operation [Gaussian, adaptive [bestfitness]])

Fig. 12. Example of configurations for *sequential partial*

V. IMPLEMENTATION REVIEW

The purpose of this section is to show that many LLHs for PSO-GA use methods that fit into the proposed taxonomy. Therefore, some LLH techniques are reviewed based on the published literature in Google Scholar and Elsevier databases. Since the reported works in LLH are very progressing, only the most current works (years of 2010-2013) are taken. Table I and Table II at the Appendix part, list the related works for *sequential* and *parallel* models respectively. The *configurations* part is not completely listed depending on the information given in each reference.

According to Table I, implementation model with *sequential global* appears to be the most popular. This is regarding to the less complexity of *sequential global model* which is not involving search space and problem distributions. Furthermore, as shown in the configurations lists, m*utation* operations have been widely included into PSO. In GA, *mutation* operation is generally thought to enable high exploration [5] whereas both exploratory and exploitative aspects are ascribed to *crossover*. Since PSO is mostly fine with exploitation, the inclusion of *mutation* is likely more essential than *crossover*.

Compared to s*equential* implementations, relatively few approaches are found for the *parallel* LLH as depicted in Table II. Prior literature studies in [9] and [12] also gained less number of implementations with *parallel* methods although for general meta-heuristics hybridizations and in a more extensive scopes of publications. However, the different number of implementation have should not be used as a performance measurement. It might subjects to many other factors including the type and size of a particular problem.

VI. CONCLUSION

Taxonomy provides general and simplified descriptions for different kinds of meta-heuristics hybridization techniques. This paper introduced taxonomy specifically for LLH of PSO-GA. Based on the taxonomy, it is also illustrates the compositions of LLH implementation models. Nevertheless, the implementation models have several limitations that demand for many research extension. A possible future research direction for improving the models can engage in handling optimization problems with multiple objectives and solutions constraints. Besides, another interesting study is to tackle two-ways LLH for the different hybrid algorithms.

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

APPENDIX

TABLE I
SEQUENTIAL MODELS

Ref.	Implementation model	Configurations
[3]	Sequential explicit global	Update (inertia[const],personal learning [const], social learning [const]) Selection (random) Mutation (rate [const], operation[adaptive[population size]])
[4]	Sequential global	Mutation (rate [const], operation [Gaussian , Adaptive[personal fitness, global fitness]) Update (Adaptive[personal fitness, global fitness],personal learning [const], social learning [const])
[5]	Sequential global	Update (inertia[const]],personal learning [const], social learning [const]) Mutation (rate [const], operation[adaptive[distance]))
[22]	Sequential global	Mutation (rate [const], operation [Gaussian, adaptive [best fitness]]) Update (inertia [adaptive [best fitness]), personal learning [const], social learning [const])
[23]	Sequential global	Update (inertia[time-vary],personal learning [const], social learning [const]) Mutation (rate [time-vary], operation [Cauchy])
[24]	Sequential global	Crossover (rate [const], operation [adaptive[personal best], time-vary [modulus]) Update (const]), personal learning [const], social learning [const])
[25]	Sequential global	Selection (Random) Mutation (rate [const], operation [Real value, Random)) Update (inertia [const], personal learning [const], social learning [const])
[26]	1.Sequential partial 2.Sequential global	1. Selection (Elite) Crossover (rate [const], operation [Uniform,[personal best],Adaptive [fitnessvalue]) Mutation (rate [const], operation [Random, Random)) Update (basic[const], personal learning [const], social learning [const])
[27]	Sequential global	Update (inertia[const],personal learning [const], social learning [const]) Mutation (rate [const], operation [Cauchy))
[28]	Sequential explicit global	Update (inertia[time-vary],personal learning [const], social learning [const]) Mutation (rate [const], operation [non-uniform)) Mutation (rate [const], operation [Cauchy))
[29]	Sequential implicit global	Update (inertia[const],personal learning [const], social learning [const]) Selection (proportional) Crossover (rate [const], operation [one-point, random)) Mutation (rate [const], operation[random))
[30]	Sequential global	Update (constriction[const],personal learning [const], social learning [const]) Selection (random) Mutation (rate [const], operation[random))
[31]	Sequential explicit global	Update (inertia[adaptive],personal learning [const], social learning [const]) Selection (roulette wheel) Crossover (rate [const], operation [adaptive[])) Mutation (rate [const], operation[adaptive[]))
[32]	Sequential global	Update (inertia[time-vary],personal learning [const], social learning [const]) Mutation (rate [const], operation[Gaussian,adaptive[]))
[33]	Sequential explicit global	Update (inertia[time-vary[LD]],personal learning [const], social learning [const]) Selection (tournament) Mutation (rate [const])
[34]	Sequential explicit global	Update (inertia[time-vary[LD]],personal learning [const], social learning [const]) Selection (random) Crossover (rate [const])
[35]	Sequential global	Update (inertia[time-vary[LD]],personal learning [const], social learning [const]) Selection (Elite) Crossover (rate [const]) Mutation (rate [const])
[36]	Sequential implicit global	Update (inertia[time-vary[LD]],personal learning [const], social learning [const]) Crossover (rate [const]) Mutation (rate [const])

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE II
PARALLEL MODELS

Ref.	Implementation model	Configurations
[37]	Parallel implicit global	- Search space (parallel [2, 0.3/0.5/0.7]) -Update (inertia [const], personal learning [const], social learning [const]) -Selection (Tournament) -Mutation (rate [const], operation [Real value, Random)) -Crossover (rate [const], operation [Random])
[38]	Parallel explicit global	- Search space (parallel [2, 0.4]) -Update (constriction [const], personal learning [const], social learning [const]) -Selection (Elite) -Mutation (rate [const], operation [Real value, Random)) -Crossover (rate [const], operation [Random])
[39]	Parallel explicit global	- Search space (parallel [2, 0.5) -Update (inertia [const], personal learning [const], social learning [const]) -Selection (Elite) -Crossover (rate [const], operation [Random]) - Mutation (rate [const], operation [Random])

REFERENCES

[1] H. Xiaohui, S. Yuhui, and R. Eberhart, "Recent advances in particle swarm," in *Congress on Evolutionary Computation, 2004. CEC2004.*, 2004, vol. 1, pp. 90-97 Vol.1.

[2] M. Affenzeller, S. Winkler, S. Wagner, and A. Beham, *Genetic Algorithms and Genetic Programming - Modern Concepts and Practical Applications.* CRC Press, 2009.

[3] C.-Y. Chen, K.-C. Chang, and S.-H. Ho, "Improved framework for particle swarm optimization : Swarm intelligence with diversity-guided random walking," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12214-12220, Sep. 2011.

[4] Q. Wu, "Power load forecasts based on hybrid PSO with Gaussian and adaptive mutation and Wv-SVM," *Expert Systems with Applications*, vol. 37, no. 1, pp. 194-201, Jan. 2010.

[5] H. Gao and W. Xu, "Particle swarm algorithm with hybrid mutation strategy," *Applied Soft Computing*, vol. 11, no. 8, pp. 5129-5142, Dec. 2011.

[6] C. Blum and A. Roli, "Hybrid Metaheuristics: An Introduction," in *Hybrid Metaheuristics*, vol. 114, C. Blum, M. J. B. Aguilera, A. Roli, and M. Sampels, Eds. Springer, 2008, pp. 1-30.

[7] G. R. Raidl, J. Puchinger, and C. Blum, "Metaheuristic Hybrids," in *Handbook of Metaheuristics*, vol. 45, M. Pardalos, H. Panos, P. Van, and M. Milano, Eds. Springer New York, 2010, pp. 305-335.

[8] S. Dower and C. J. Woodward, "ESDL : A Simple Description Language for Population-Based Evolutionary Computation," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation (GECCO '11)*, 2011, pp. 1045-1052.

[9] E. G. Talbi, "A Taxonomy of Hybrid Metaheuristics," *Jounal of Heuristics*, vol. 8, pp. 541-564, 2002.

[10] C. Blum, A. Roli, and E. Alba, "An Introduction to Metaheuristic Techniques," in *Parallel Metaheuristic: A New Class of Algorithms*, E. Alba, Ed. John Wiley and Sons, 2005.

[11] E. G. Talbi, *Metaheuristics: From Design to Implementation*. Wiley, 2009, p. 586.

[12] S. Masrom, S. Z. . Abidin, and N. Omar, "A Taxonomy of Low-level Hybridization in Metaheuristics Algorithms," in *IEEE fifth International Conference on Advance Computational Intelligence (ICACI)*, 2012, pp. 435-440.

[13] T. G. Crainic and M. Toulouse, "Parallel Strategies for Meta-Heuristics ," in *Handbook of Metaheuristics*, vol. 57, F. Glover and G. A. Kochenberger, Eds. Springer New York, 2003, pp. 475-513.

[14] M. El-Abd and M. Kamel, "A Taxonomy of Coorperative Search Algorithm," in *Hybrid Metaheuristic 2005*, vol. 3636, M. J. B. et al., Ed. Springer-Verlag Berlin Heidelberg, 2005, pp. 32-41.

[15] T. G. Crainic and M. T. Gendreau, "Towards a Taxonomy of Parallel Tabu Search Heuristics," *INFORMS Journal of Computing*, vol. 9, no. 1, pp. 61-72, 1997.

[16] M. El-abd and M. S. Kamel, "A taxonmy of cooperative particle swarm optmization," *Computational Intelligence*, vol. 4, no. 2, pp. 137-144, 2008.

[17] C. García-Martínez, O. Cordón, and F. Herrera, "A taxonomy and emirical analysis of multiple objective ant colony optimizatin algorithms for the bi-criteria TSP," *European Journal of Operational Research*, vol. 180, no. 1, pp. 116-148, 2007.

[18] F. Herrera, M. Lozano, and A. M. Sánchez, "A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study," *International Journal of Intelligent Systems*, vol. 18, no. 3, pp. 309-338, 2003.

[19] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy, and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 474-488, 2005.

[20] F. Herrera and M. Lozano, "Fuzzy adaptive genetic algorithms: design, taxonomy and future directions," *Soft computing*, vol. 7, no. 545-562, 2003.

[21] H. C. Lau, W. C. Wan, S. Halim, and K. Toh, "A software framework for fast prototyping of meta-heuristics hybridization," *International Transactions in Operational Research*, vol. 14(2), pp. 123-141, 2007.

[22] A. Alireza, "PSO with Adaptive Mutation and Inertia Weight and Its Application in Parameter Estimation of Dynamic Systems," *Acta Automatica Sinica*, vol. 37, no. 5, pp. 541-549, 2011.

[23] S. Masrom, S. Z. Z. Abidin, N. Omar, and K. Nasir, "Time-Varying Mutation in Particle Swarm Optimization," in *LNAI*, Springer-Verlag Berlin Heidelberg, 2013, pp. 31-40.

[24] S. Chen, "Particle swarm optimization with pbest crossover," *2012 IEEE Congress on Evolutionary Computation*, pp. 1-6, Jun. 2012.

[25] H. Lu, P. Sriyanyong, Y. H. Song, and T. Dillon, "Experimental study of a new hybrid PSO with mutation for economic dispatch with non-smooth cost function," *International Journal of Electrical Power & Energy Systems*, vol. 32, no. 9, pp. 921-935, 2010.

[26] R. J. Kuo and Y. S. Han, "A hybrid of genetic algorithm and particle swarm optimization for solving bi-level linear programming problem - A case study on supply chain model," *Applied Mathematical Modelling*, vol. 35, pp. 3905-3917, 2011.

[27] S.Masrom, Siti.Z.Z.Abidin, A.M.Nasir, and A.S.A.Rahman, "Hybrid Particle Swarm Optimization for Vehicle Routing Problem with Time Windows," in *International Conference on Recent Researches in Computational Techniques, Non-Linear Systems and Control*, 2011, pp. 142-147.

[28] M. Hu, T. Wu, and D. Weir, "An intelligent augmentatio of particle swarm optimization with multiple adaptive methods," *Information Sciences*, vol. 213, pp. 68-63, 2012.

[29] H.-J. Lin and J. P. Yeh, "A hybrid optimization strategy for simplifying the solutions of support vector machines," *Pattern Recognition Letters*, vol. 31, pp. 563-571, 2010.

[30] Y. Zhou and Y. Tan, "Particle swarm optimization with triggered mutation and its implementation based on GPU," *Proceedings of the*

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

12th annual conference on Genetic and evolutionary computation - GECCO '10, pp. 1-8, 2010.

[31] Y. Marinakis and M. Marinaki, "A hybrid genetic - Particle Swarm Optimization Algorithm for the vehicle routing problem," *Expert System with Applications*, vol. 37, pp. 1446-1455, 2010.

[32] Q. Wu and R. Law, "Cauchy mutation based on objective variable of Gaussian particle swarm optimization for parameters selection of SVM," *Expert Systems with Applications*, vol. 38, no. 6, pp. 6405-6411, Jun. 2011.

[33] M. Gang, Z. Wei, and C. Xiaolin, "A novel particle swarm optimization algorithm based on particle migration," *Applied Mathematics and Computation*, vol. 218, no. 11, pp. 6620-6626, Feb. 2012.

[34] M. Li, D. Lin, and J. Kou, "A hybrid niching PSO enhanced with recombination-replacement crowding strategy for multimodal function optimization," *Applied Soft Computing*, vol. 12, no. 3, pp. 975-987, Mar. 2012.

[35] Y. Shao, Q. Chen, and C. Li, "Research on Hybrid Improved PSO Algorithm," in *ISICA 2010*, Z. Cai et ., Springer-Verlag Berlin Heidelberg, 2010, pp. 234-242.

[36] R. J. Kuo, F. E. Zulvia, and K. Suryadi, "Hybrid particle swarm optimization with genetic algorithm for solving capacitated vehicle routing problem with fuzzy demand – A case study on garbage collection system," *Applied Mathematics and Computation*, vol. 219, no. 5, pp. 2574-2588, Nov. 2012.

[37] W.-P. Lee and Y.-T. Hsiao, "Inferring gene regulatory networks using a hybrid GA-PSO approach with numerical constraints and network decomposition," *Information Sciences*, vol. 188, pp. 80-99, 2012.

[38] S. Yu, Y.-M. Wei, and K. Wang, "A PSO-GA optimal model to estimate primary energy demand of China," *Energy Policy*, vol. 42, pp. 329-34-, 2012.

[39] B. Mhamdi, G. Khaled, and T. Aguili, "Hybrid of genetic algorithm with particle swarm optimization to shape reconstruction of perfect conducting cylinders," *International Journal of Electronics and Communications*, vol. 65, pp. 1032-1039, 2011.

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)